Mean-payoff objectives for Markov Decision Processes

On Value Iteration

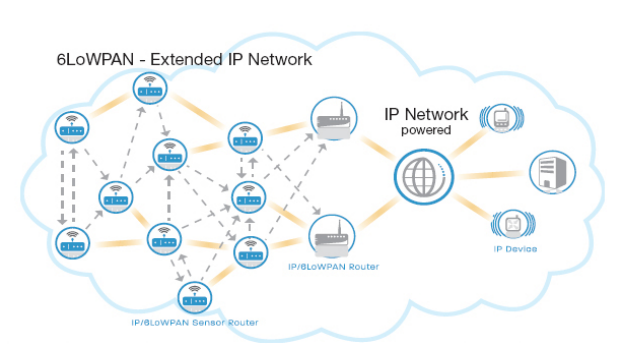
Based on a paper accepted at CAV 2017

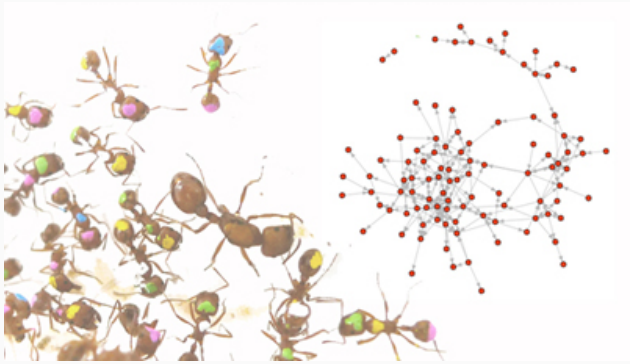
Pranav Ashok[1], Krishnendu Chatterjee[2], Przemysław Daca[2],
Jan Křetínský[1] and Tobias Meggendorfer[1]

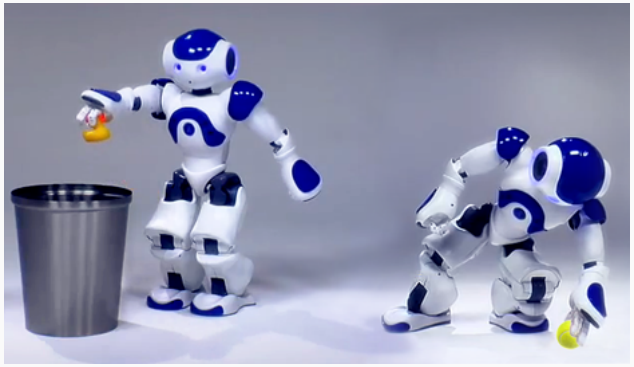
April 23, 2017

[1]Technical University of Munich, Germany
[2]IST Austria

Motivation

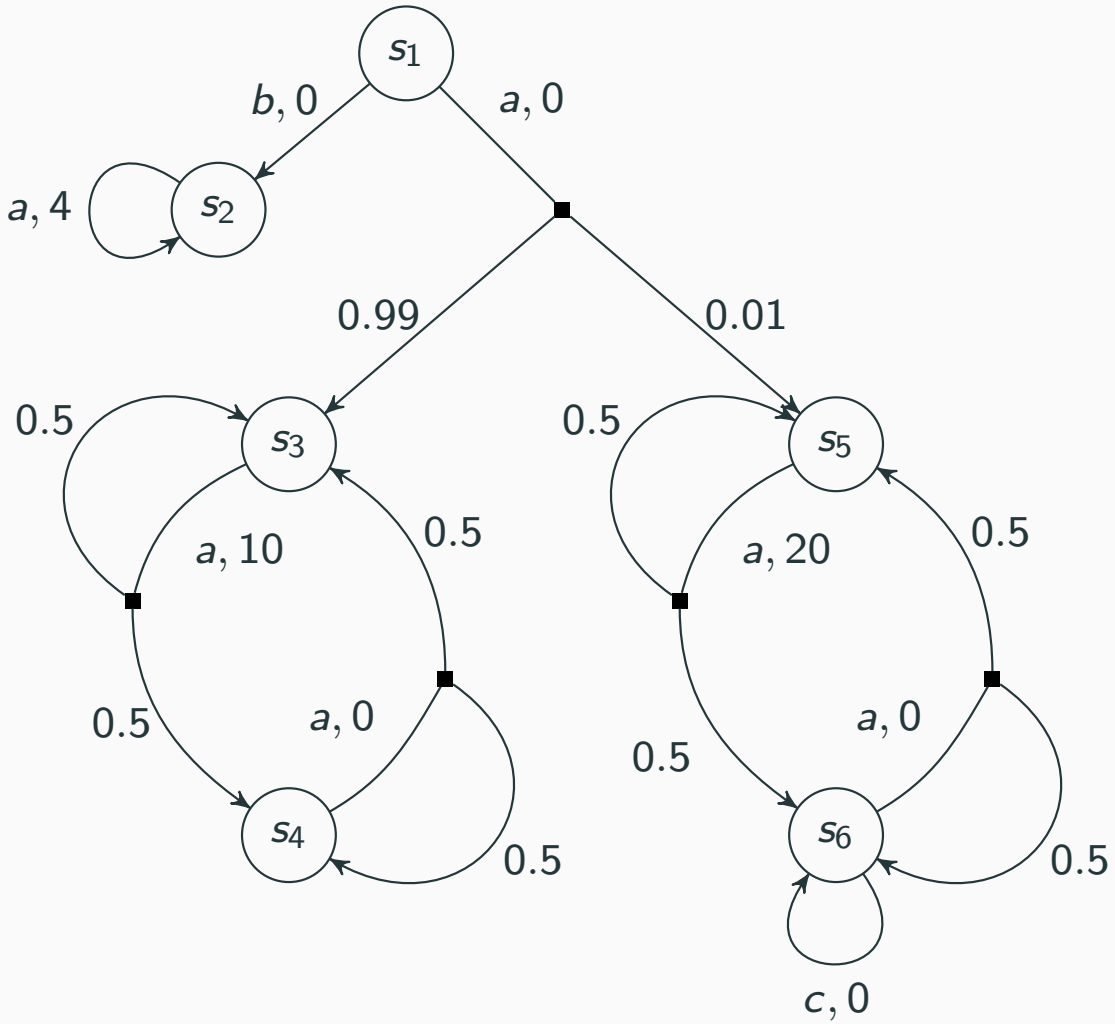






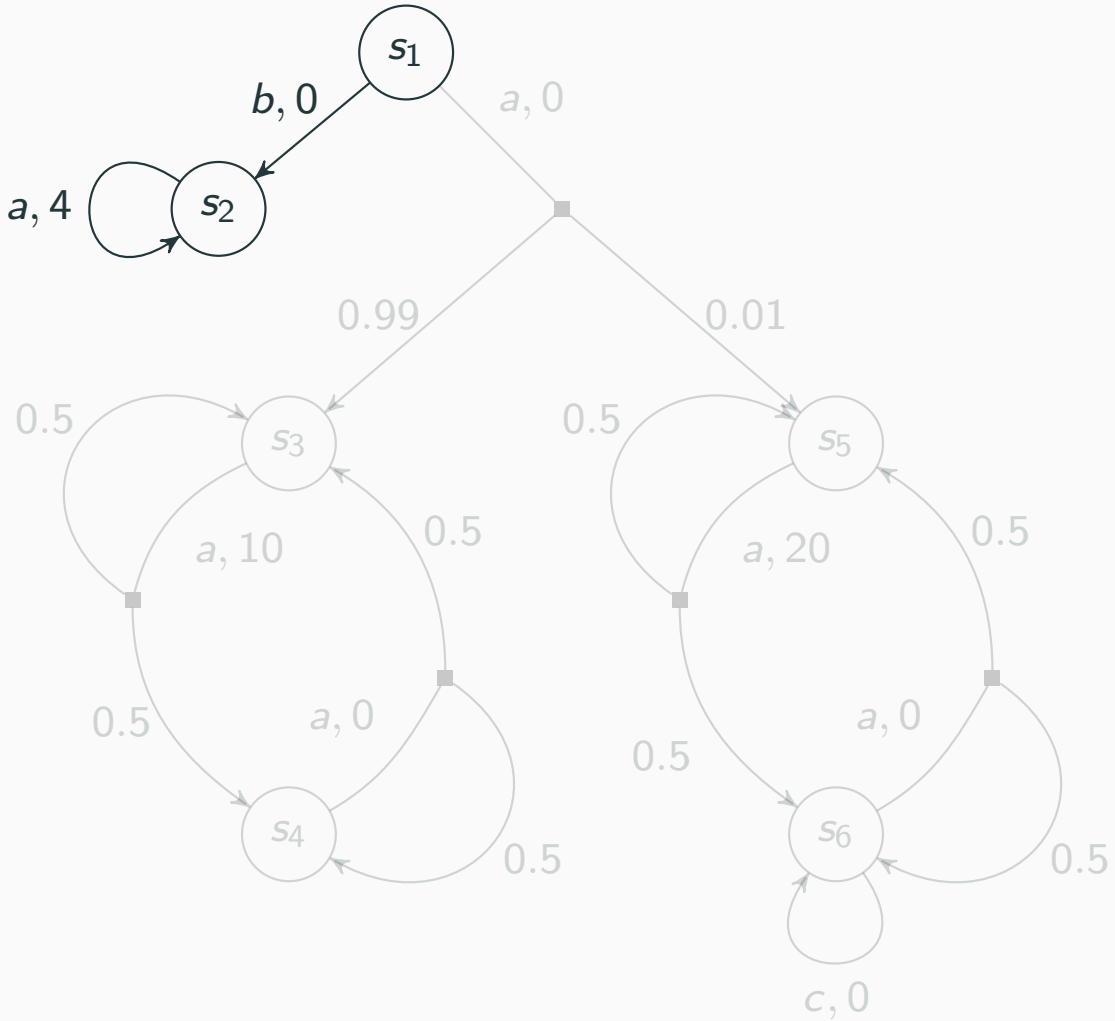
Markov Decision Processes (MDPs) are a standard model for describing systems which display probabilistic as well as non-deterministic behaviour.

Markov Decision Process (MDPs)



Strategy

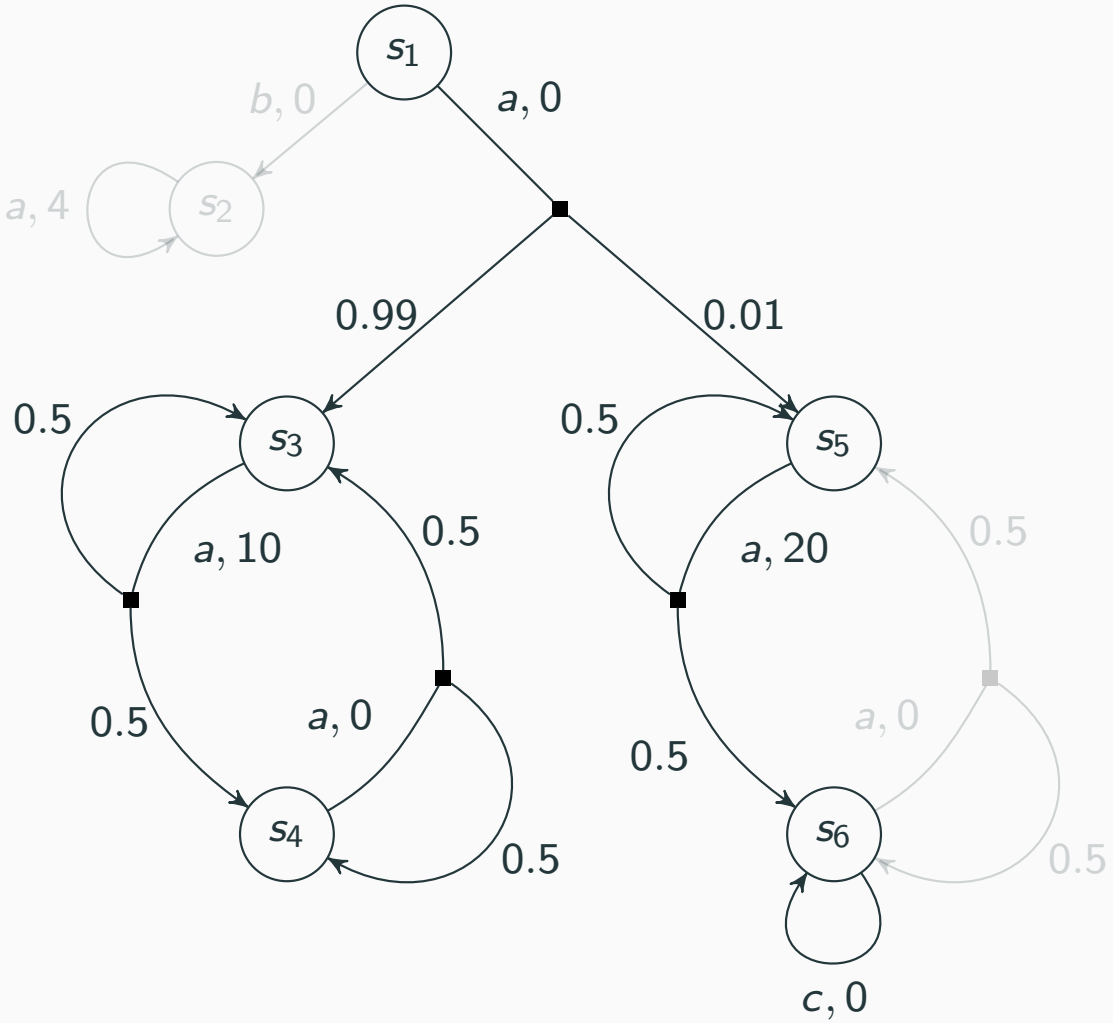
A strategy (or policy) gives the action to be taken at every state.[1]



[1] We are only representing partial strategies in the diagrams

Strategy

A strategy (or policy) gives the action to be taken at every state.[1]



[1] We are only representing partial strategies in the diagrams

Mean-payoff

Mean payoff or Long-run average reward

$$\rho = 0 \;\; 10 \;\; 10 \;\; 10 \;\; 0 \;\; 10 \;\; 0 \;\; 10 \;\; 0 \;\; 10 \;\; 0 \ldots$$

Then, n-step average reward is given by

$$MP_n(\rho) \;:=\; \frac{1}{n} \cdot \sum_{i=1}^{n} \rho_i \quad \xrightarrow{n \to \infty} \quad 5$$

We usually talk about min/max mean-payoff.

$$v := \sup_{\pi} \lim_{n \to \infty} \mathbb{E}^{\pi}[MP_n(\rho)]$$

Algorithms[2]

- Linear Programming (LP)
- Strategy Iteration (SI)
- Value Iteration (VI)

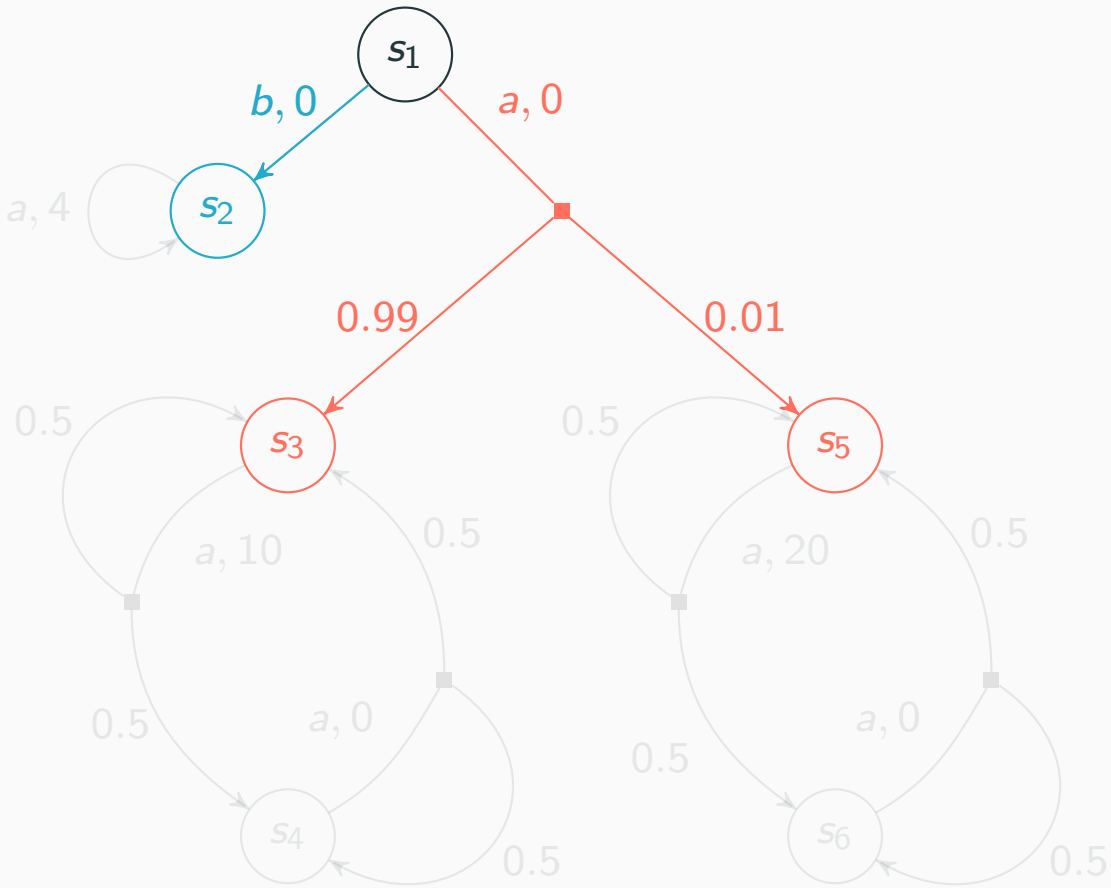
[2] Markov Decision Processes, Martin L. Puterman, '94

Value Iteration

Total Expected Rewards

$$v_n(s_1, a) = \{0 + 0.99 \cdot v_{n-1}(s_3) + 0.01 \cdot v_{n-1}(s_5)\}$$
$$v_n(s_1, b) = \{0 + v_{n-1}(s_2)\}$$

} max



Value Iteration

Total reward

$$v_n(s,a) = r(s,a) + \sum_{s'} p(s,a,s')v_{n-1}(s')$$

$$v_n(s) = \max_a \{v_n(s,a)\}$$

Average reward

$$\lim_{n\to\infty} \frac{v_n(s)}{n} \approx (v_n(s) - v_{n-1}(s)), \text{ for large } n$$

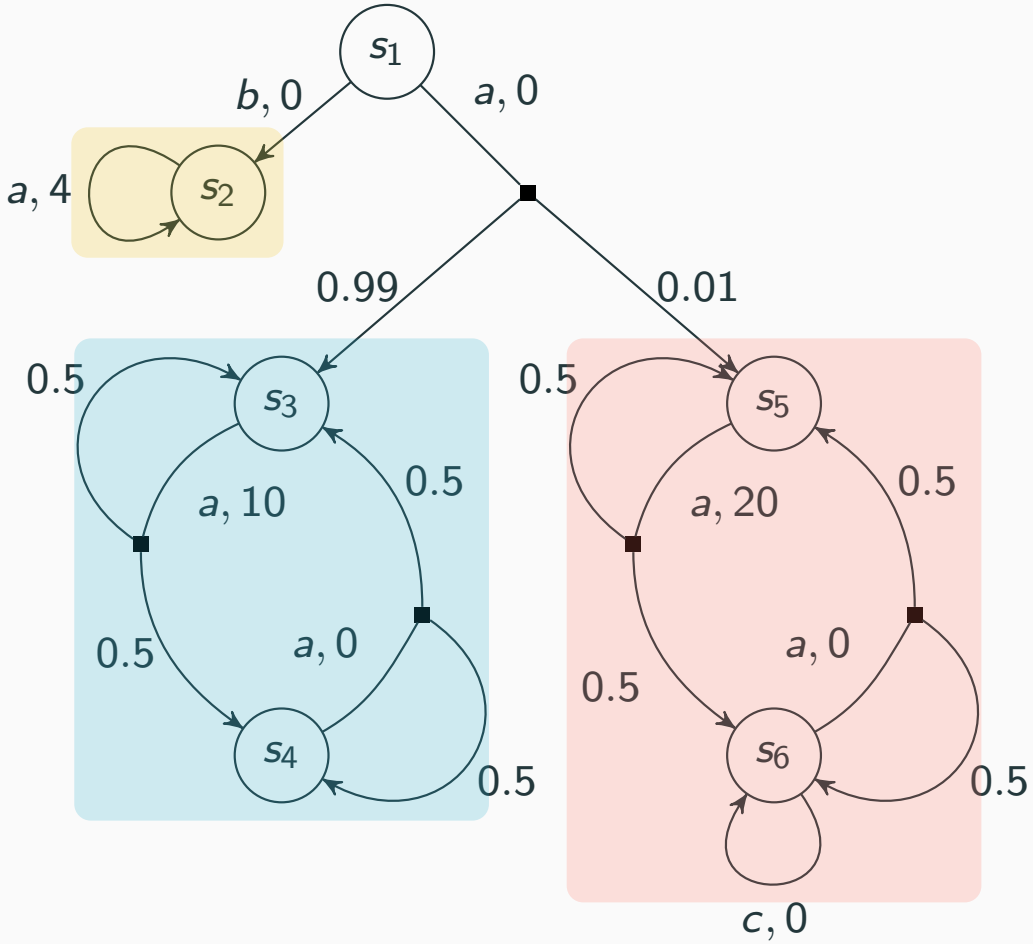
Our Contribution

Computing ε-optimal solutions

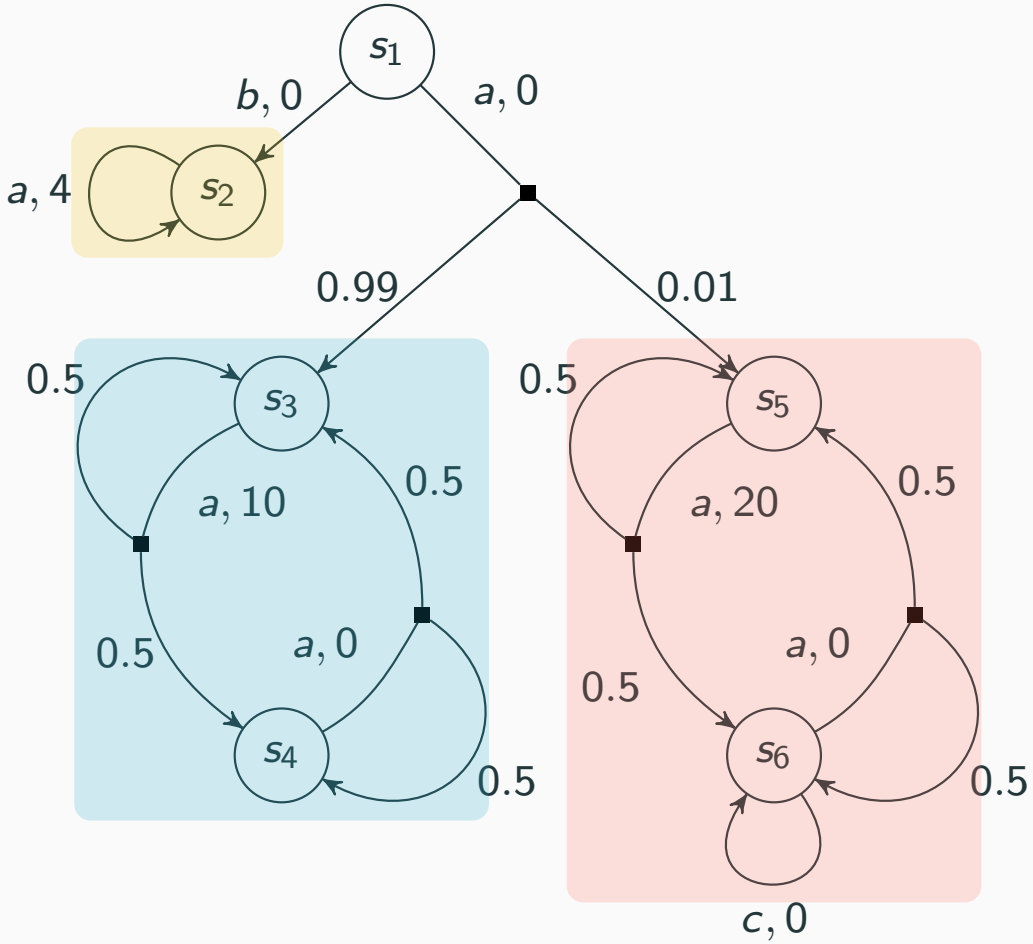
Goals:

- Identify what contributes the most to the mean-payoff and exploit this knowledge
- Use techniques from machine learning to avoid exploring the whole MDP

Another look at the MDP



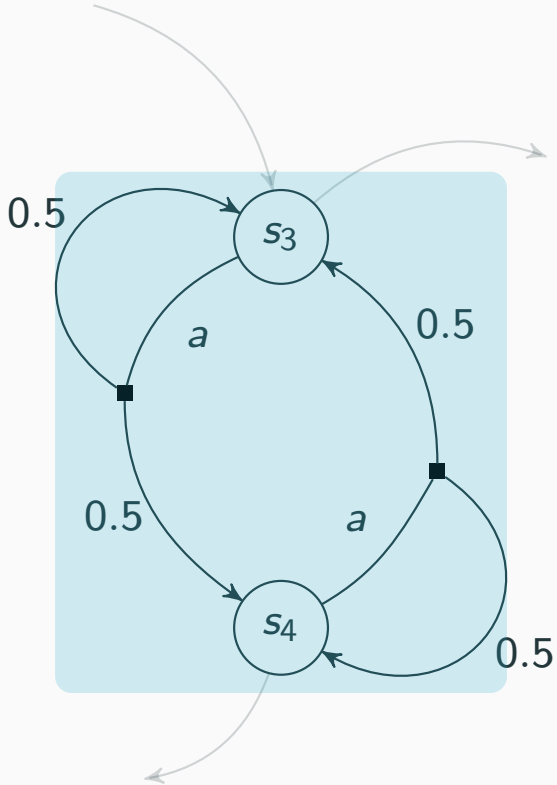
Another look at the MDP



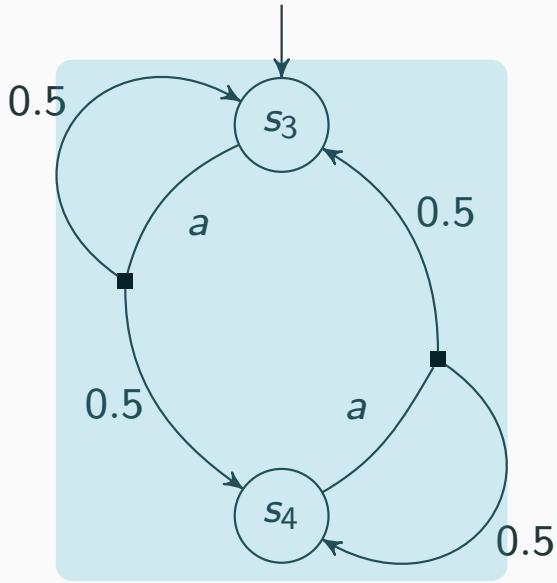
Maximal end-components!

Maximal End Components (MECs)

A MEC is a (maximal) pair (S', A') such that
- successors along every action in A' is part of S'
- for all $s_x, s_y \in S'$ there exists $s_x \xrightarrow{*} s_y$.



An example MEC



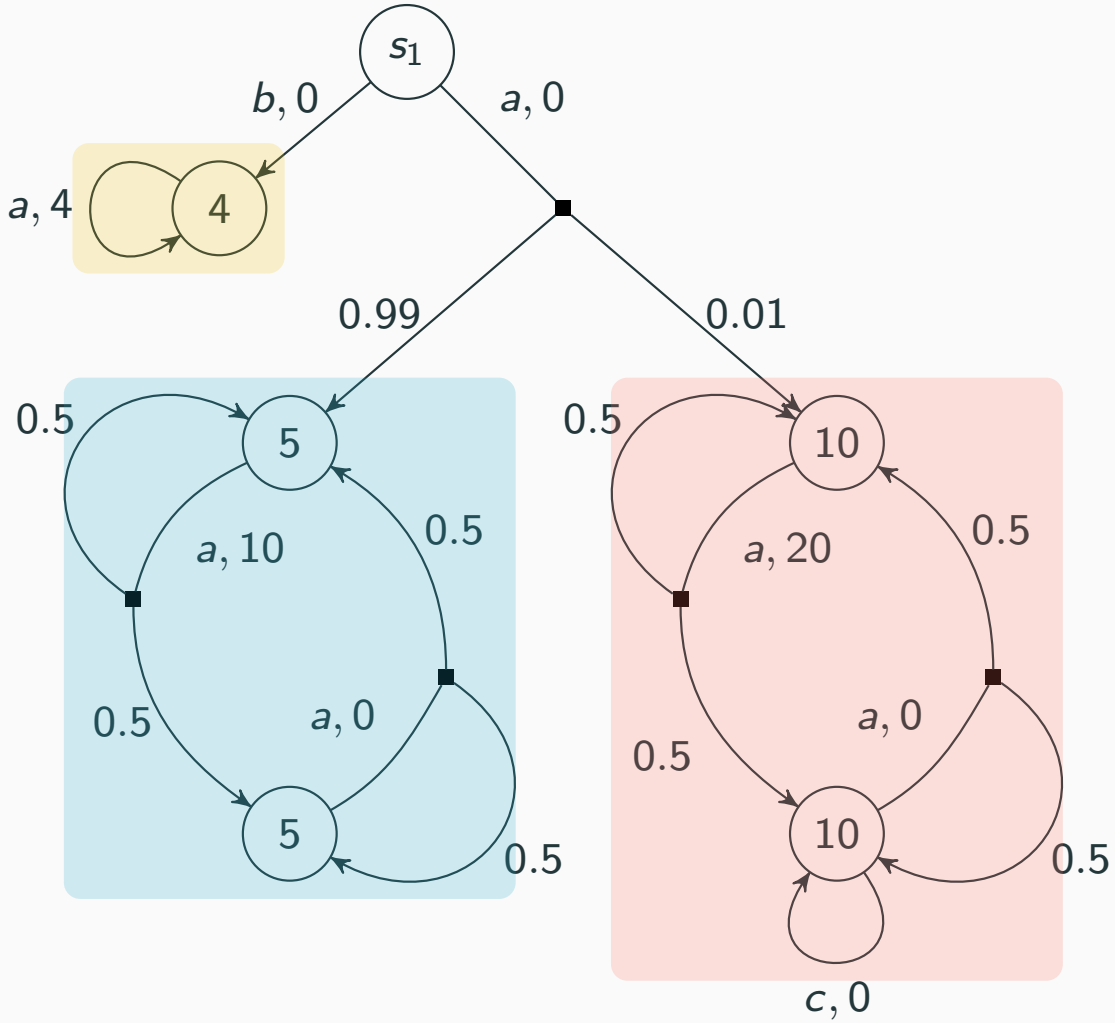
MEC viewed as an MDP

Recap

- In general, not known when to stop VI so as to get ϵ-optimal value
- But known for MDPs in which every state can reach every other state via some action (communicating MDPs)[3]
- Moreover, only MECs contribute to the mean-payoff

[3] Martin L. Puterman, Markov Decision Processes, 1994

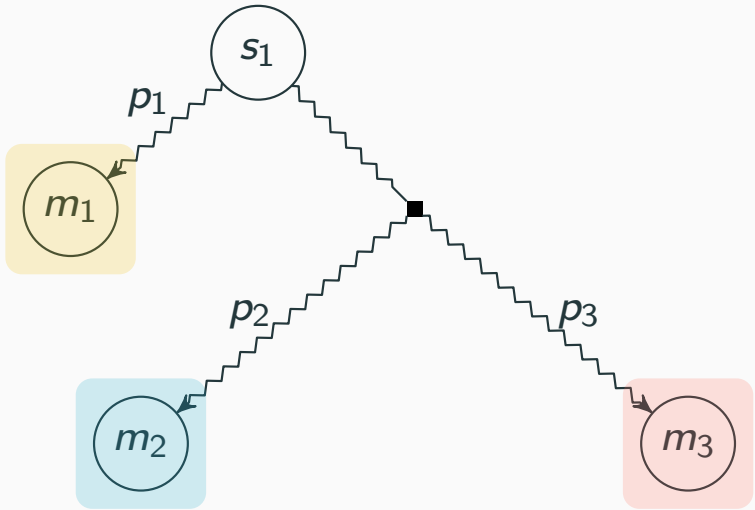
Approach (MEC-Decomposition)



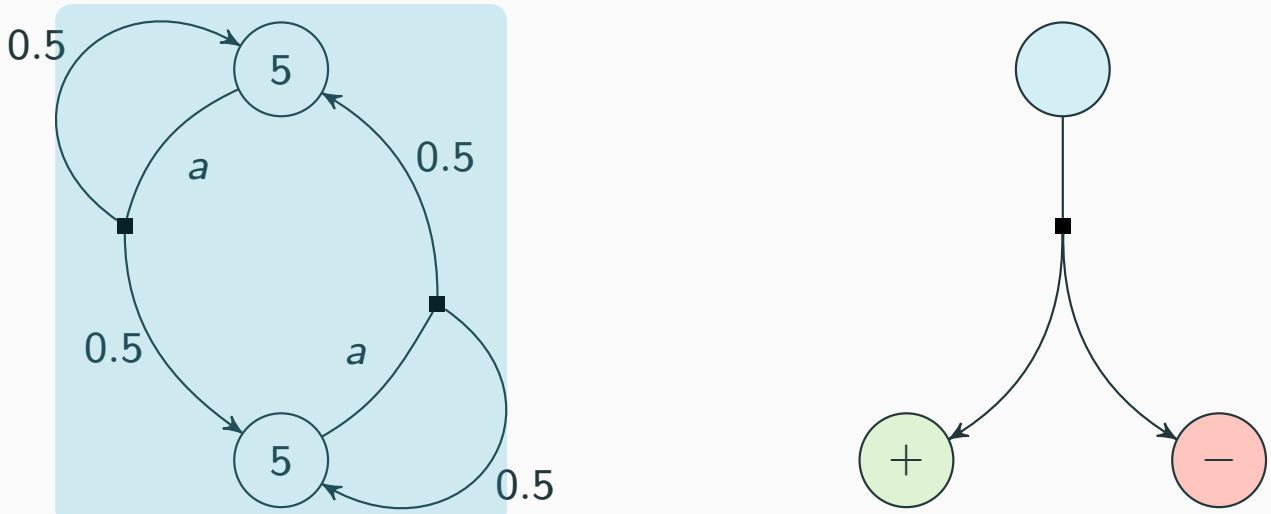
Find all MECs and perform VI on them until ε-convergence

Transformation to reachability problem

$$\text{Max. mean-payoff} = \sup_{\pi} p_1 m_1 + p_2 m_2 + p_3 m_3$$



MEC Collapsing

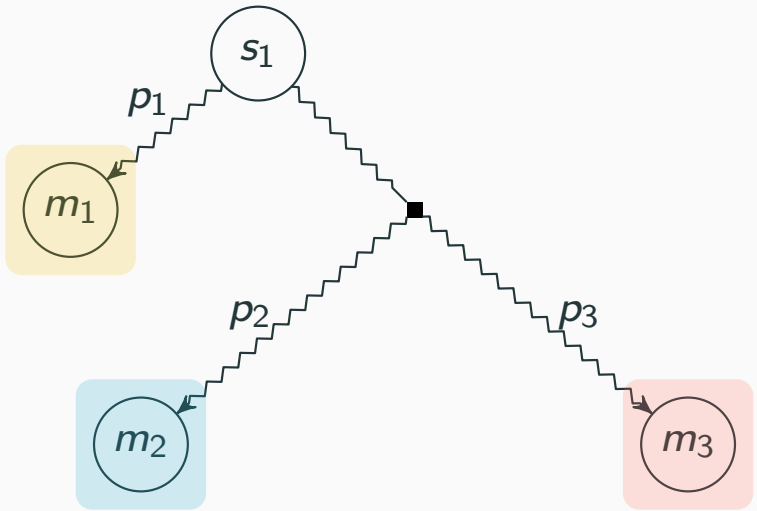


Collapse MEC into a single state and add a special action

Transformation to reachability problem

Let the largest MEC reward be $R = max\{m_1, m_2, m_3\}$. Then,

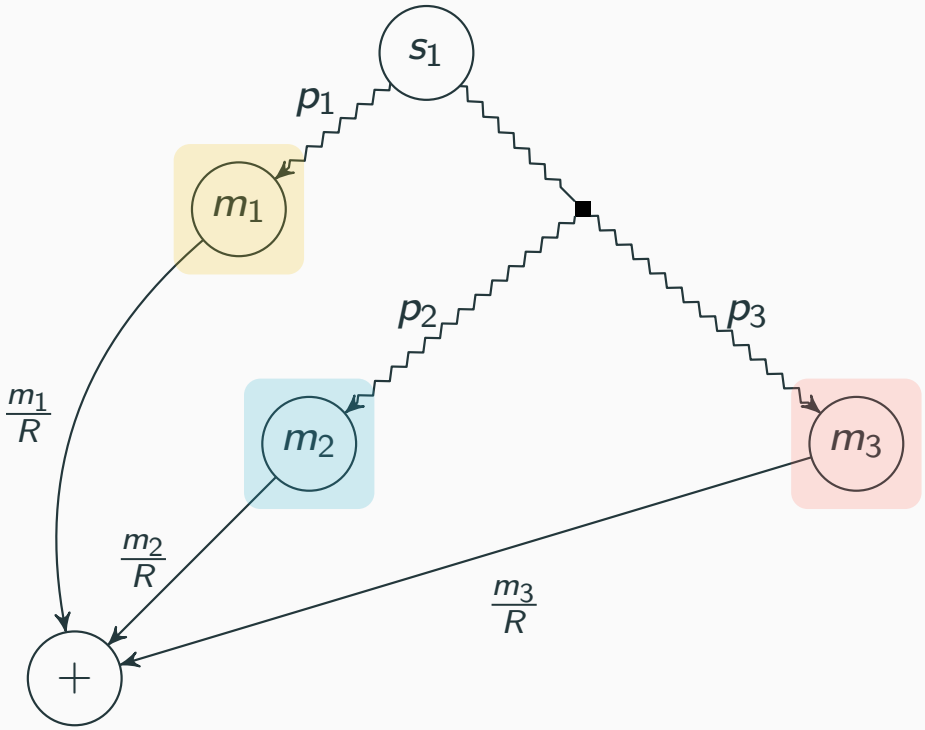
$$\frac{\text{Max. mean-payoff}}{R} = \sup_{\pi} p_1 \frac{m_1}{R} + p_2 \frac{m_2}{R} + p_3 \frac{m_3}{R}$$



Transformation to reachability problem

Let the largest MEC reward be $R = max\{m_1, m_2, m_3\}$. Then,

$$\frac{\text{Max. mean-payoff}}{R} = \sup_{\pi} p_1 \frac{m_1}{R} + p_2 \frac{m_2}{R} + p_3 \frac{m_3}{R}$$

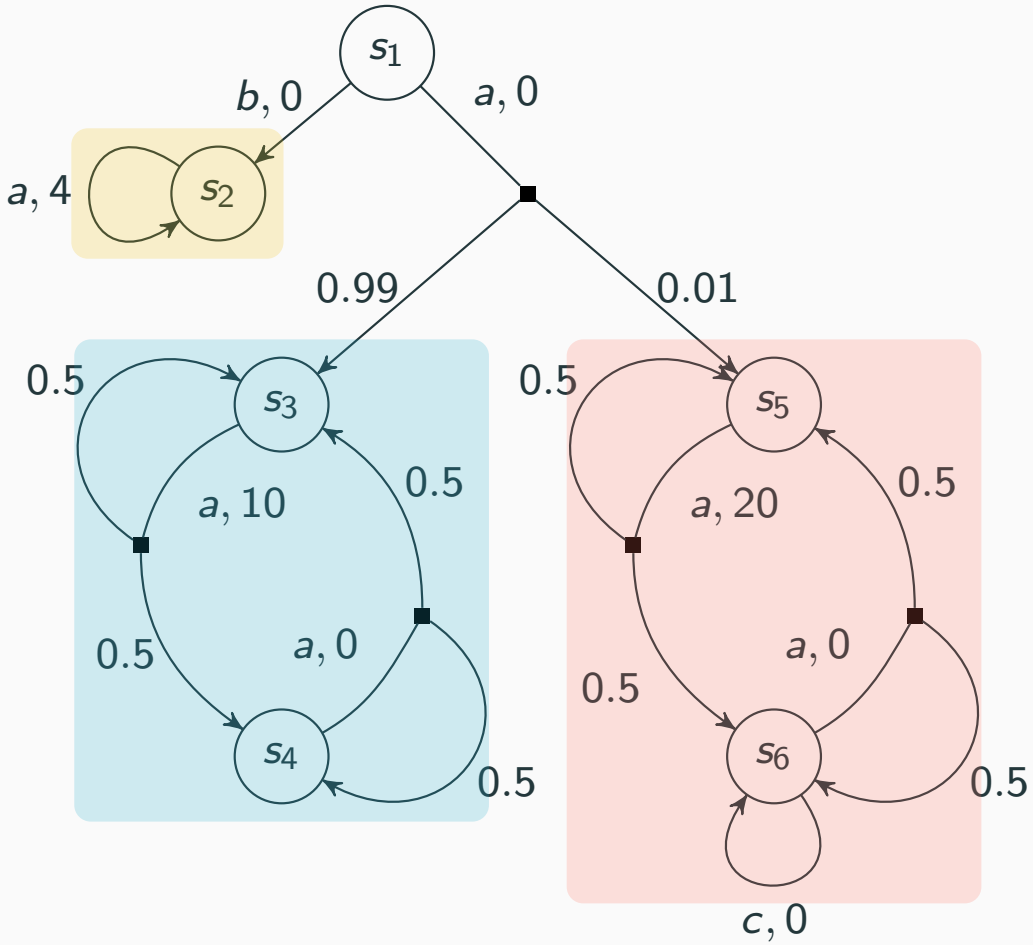


Can we do better?

- Whole state space is being explored
- Finding MECs on large state spaces is computationally expensive

Heuristic Method 1

Idea: Partial exploration using sampling – let sampling guide us to the "important" regions



Red region is reachable with small probability

Heuristic Method 1

- Existing: BRTDP approach for reachability[4][5]
- Maintains lower and upper bounds for every state, L and U
- Uncertainty is given by $(U - L)$
- Target state: U = L = 1; Sink states: U = L = 0
- Repeatedly samples paths from initial state
- Back-propagates values along the path using VI operator

[4] Bounded Real-Time Dynamic Programming, McMahan et. al., ICML '05

[5] Verification of Markov Decision Processes using Learning Algorithms, Brazdil et. al., ATVA '14

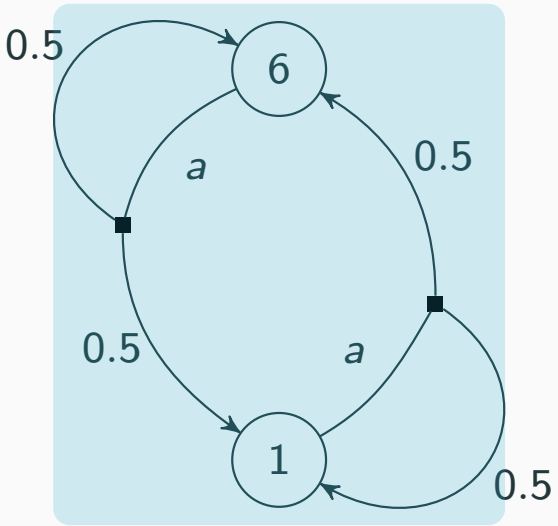
Heuristic Method 1

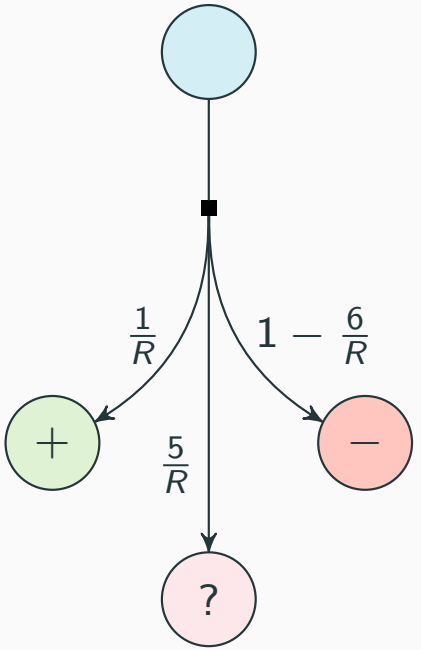
- Repeatedly collapse MECs amongst the states explored so far
- While collapsing, compute their values and add transition to the special $+$ and $-$ states
- Solve reachability for $+$

Heuristic Method 2

Even better?

- Sample paths like in the previous method
- When MECs are detected, collapse them but...
- Don't compute MEC value until convergence
- Add transition with probability $\propto$ (U-L) to a **?** state
- Refine value of a MEC only when the **?** state is encountered





Collapse MEC into a single state and add a special action

Summary

We saw two methods which can be used to obtain mean-payoff with guarantees

1. Collapse MECs, add transitions to $+/-$ states, run reachability
2. Run sampling, collapse on-the-go, refine MEC values only when **?** is encountered

Benchmarks

Model	States	MECs	LP[1]	MEC-VI[2]	MEC-BRTDP[3]	HM2[4]
virus	809	1	0.19	**0.05**	0.05	TO
cs_nfail4	960	176	0.7	**0.18**	0.19	3.34
investor	6 688	837	2.8	0.51	**0.47**	1.32
phil-nofair5	93 068	1	TO	**6.67**	6.92	TO
rabin4	668 836	1	TO	**112.38**	112.49	TO

1. MultiGain, Brazdil et. al. 2015.
2. MEC-VI: Straightforward conversion to reachability, then VI
3. MEC-BRTDP: Straightforward conversion to reachability, then BRTDP
4. HM2: Heuristic Method 2, refine MEC values when needed

Benchmarks

Heuristic Method 2 better by orders of magnitude depending on topology

Model	**States**	`MEC-VI`	`HM2`	`HM2` States	`HM2` MECs
zeroconf(40,10)	3 001 911	MO	5.05	481	3
avoid				582	3
zeroconf(300,15)	4 730 203	MO	16.6	873	3
avoid				5 434	3
sensors(2)	7 860	18.9	20.1	3 281	917
sensors(3)	77 766	2293.0	37.0	10 941	2 301

Conclusions

- If the MDP is a single MEC, then the standard VI[6], works the best.
- If the MDP is not a single MEC, then
    1. for small models, the MEC-collapsing methods work the fastest
    2. for large models, usually Heuristic Method 2 works the best

[6] Markov Decision Processes, Martin L. Puterman, '94